<Priority Document Translation>

THE KOREAN INDUSTRIAL PROPERTY OFFICE

This is to certify that annexed hereto is a true copy from the records of the Korean Industrial Property Office of the following application as filed.

Application Number : 2000-83269 (Patent)

Date of Application : December 27, 2000

Applicant(s) : ELECTRONICS & TELECOMMUNICATION RESEARCH INSTITUTE

March 2, 2001

COMMISSIONER

CERTIFIED COPY OF
PRIORITY DOCUMENT

대 한 민 국 특 허 청

KOREAN INDUSTRIAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Industrial Property Office.

출 원 번 호 : 특허출원 2000년 제 83269 호
Application Number

출 원 년 월 일 : 2000년 12월 27일
Date of Application

출 원 인 : 한국전자통신연구원
Applicant(s)

2001 년 03 월 02 일

특 허 청
COMMISSIONER

【서류명】 특허출원서
【권리구분】 특허
【수신처】 특허청장
【참조번호】 0017
【제출일자】 2000.12.27
【발명의 명칭】 공유메모리 스위치에서의 멀티캐스팅 장치 및 그 방법
【발명의 영문명칭】 Multicasting apparatus and method in shared memory switch

【출원인】
【명칭】 한국전자통신연구원
【출원인코드】 3-1998-007763-8

【대리인】
【성명】 특허법인 신성 정지원
【대리인코드】 9-2000-000292-3
【포괄위임등록번호】 2000-051975-8

【대리인】
【성명】 특허법인 신성 원석희
【대리인코드】 9-1998-000444-1
【포괄위임등록번호】 2000-051975-8

【대리인】
【성명】 특허법인 신성 박해천
【대리인코드】 9-1998-000223-4
【포괄위임등록번호】 2000-051975-8

【발명자】
【성명의 국문표기】 김찬
【성명의 영문표기】 KIM,Chan
【주민등록번호】 680305-1068712
【우편번호】 305-345
【주소】 대전광역시 유성구 신성동 153번지 하나아파트 110-505
【국적】 KR

【발명자】
【성명의 국문표기】 유태환
【성명의 영문표기】 YOO,Tae Whan
【주민등록번호】 580701-1036616

【우편번호】 305-345
【주소】 대전광역시 유성구 신성동 하나아파트 106-1302
【국적】 KR
【발명자】
【성명의 국문표기】 이종현
【성명의 영문표기】 LEE,Jong Hyun
【주민등록번호】 590216-1090427
【우편번호】 305-333
【주소】 대전광역시 유성구 어은동 한빛아파트 110-504
【국적】 KR
【심사청구】 청구
【취지】 특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인
특허법인 신성 정지원 (인) 대리인
특허법인 신성 원석희 (인) 대리인
특허법인 신성 박해천 (인)
【수수료】

【기본출원료】	20	면	29,000 원
【가산출원료】	14	면	14,000 원
【우선권주장료】	0	건	0 원
【심사청구료】	11	항	461,000 원

【합계】 504,000 원
【감면사유】 정부출연연구기관
【감면후 수수료】 252,000 원
【첨부서류】 1. 요약서·명세서(도면)_1통

【요약서】

【요약】

본 발명은, 공유메모리 스위치에서의 멀티캐스팅 장치 및 그 방법과 상기 방법을 실현시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체에 관한 것으로서, 공유메모리 스위치 등에서 최대의 속도로 포인터(데이터가 저장된 주소)의 멀티캐스팅을 위하여, 공유 메모리 기반 스위치에서의 멀티캐스팅 방법에 있어서, 모든 포인터(데이터가 저장된 주소)를 입력 서브큐에 저장하였다가, 출력 포트 비트 맵과 사용되는 경우 해당 데이터의 클래스 정보에 따라 해당 출력 서브큐로 옮기고, 멀티캐스트 데이터의 경우 동일한 포인터를 다수의 출력 서브큐로 복사하여 옮기되, 상기 포인터를 상기 입력 서브큐에서 상기 해당 출력 서브큐로 옮길 때, 상기 입력 서브큐에서 읽은 데이터에 대해서 한 번에 한 비트씩 선택하여 한 비트만 선택된 비트 맵의 스트림으로 만들고, 상기 입력 서브큐에 데이터가 있는 경우 현재 처리하고 있는 데이터가 모두 처리되기 전에 미리 상기 입력 서브큐의 데이터를 읽음으로써 상기 입력 서브큐의 데이터와 데이터 사이에 대기 시간이 없도록 하는 것을 특징으로 한다.

【대표도】

도 1

【색인어】

멀티캐스팅, 포인터, 입력 서브큐, 출력 서브큐, 공유 메모리

【명세서】

【발명의 명칭】

공유메모리 스위치에서의 멀티캐스팅 장치 및 그 방법{Multicasting apparatus and method in shared memory switch}

【도면의 간단한 설명】

도 1 은 본 발명이 적용되는 공유 메모리 스위치의 포인터 처리 방식을 나타낸 설명도.

도 2 는 본 발명에 따른 멀티캐스팅 장치에 대한 일실시예 구성도.

도 3 은 본 발명에 따른 상기 도 2의 입력큐 읽기부의 일실시예 상세 구성도.

도 4a 내지 4e 는 본 발명에 따라 입력 서브큐에서 읽기 신호를 발생하는 조건 및 각 레지스터의 값을 선택하는 조건을 나타낸 일실시예 흐름도.

도 5 는 본 발명에 따른 멀티캐스트를 위해 메모리내의 카운터를 읽기 및 쓰기하는 과정을 나타낸 일실시예 설명도.

* 도면의 주요 부분에 대한 부호의 설명

10 : 입력큐 읽기부 20 : 큐번호 변환부

30 : 출력큐 쓰기부

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<9> 본 발명은 공유 메모리 방식의 비동기전달모드(ATM : Asynchronous Transfer Mode) 셀 또는 가변길이 패킷 스위치에서의 멀티캐스팅 장치 및 그 방법과 상기 방법을 실현시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체에 관한 것으로, 특히 최대의 속도로 포인터(데이터가 저장된 주소)의 멀티캐스팅을 위하여 입력 서브큐의 포인터 내용을 해당하는 여러 개의 출력 서브큐로 옮기는 것이다. 또한, 연속으로 메모리 내부의 카운터를 읽고 수정하여 쓰는 경우에 메모리 값의 일관성을 유지하고자 하는 것이다.

<10> 공유 메모리 스위치 설계시에, 데이터를 저장하기 위한 메모리는 자원으로서 관리하기 위해 작은 영역으로 나뉘어져서 관리된다. 이때, 데이터가 수신되어 저장될 영역이 필요하면, 비사용 번지 리스트에서 사용되지 않는 번지를 가져와서 데이터를 보관하는데 사용하고, 이후에 스케쥴링을 거쳐 해당 번지의 데이터가 읽혀지고 나면 해당 번지가 다시 비사용 리스트로 반환된다.

<11> 멀티캐스팅(Multicasting)을 지원하는 공유 메모리에서는 멀티캐스팅 데이터이건 아니건 실제 데이터는 오직 한 번 저장된다.

<12> 멀티캐스팅을 구현하기 위한 한 가지 방법으로, 데이터가 저장된 번지(즉, 포인터)들이 일단 입력 서브큐에 저장되었다가 하나씩 꺼내어 출력 포트 및 클래스에 따라 해당하는 출력 서브큐에 전달하는 방식이 있다.

<13> 이러한 경우에 멀티캐스팅은 포인터를 여러 개의 출력 서브큐에 넣음으로써 이루어지게 된다. 이렇게 포인터를 필요로 하는 경우 복사를 하면서 이동시키는 과정은, 먼저 입력 서브큐를 읽어 보통은 비트 맵으로 표시된 목표지 포트를 읽음으로써 시작된다. 다음으로, 여러 비트중에서 한 비트씩 선택하여 해당 클래스를 고려하여 해당하는 출력 서브큐에 써 넣는 과정이 처리되는데 입력 서브큐에서 읽은 포트 비트 맵 데이터에 대해서 모든 비트가 한번씩 선택되도록 하여야 한다. 하나의 아이템이 처리되고 나면 입력 서브큐에서 다음 데이터를 읽어 처리하게 된다.

<14> 그러나, 이러한 경우에 최대의 속도로 처리하지 않는 방식에서는 포인터를 옮기는 과정에서 모든 클럭 주기를 사용하지 못하게 되고 이 부분이 바틀넥(bottle neck)이 되어 전체 시스템의 성능을 떨어뜨리게 된다.

<15> 따라서, 이러한 경우에 한 클럭도 잃어 버리지 않고 매번 출력 서브큐로 포인터를 옮길 수 있는 방안이 필수적으로 요구된다.

<16> 한편, 멀티캐스팅을 구현하기 위한 다른 방법으로, 포인터를 여러 개의 출력 서브큐에 삽입하는 경우에, 각 출력 포트 처리기는 자신의 출력 서브큐를 읽어 해당 포인터의 공유 메모리 번지에서 데이터를 읽게 되는데 멀티캐스팅의 수만큼 데이터를 여러 번 읽은 후에 해당 번지를 비사용 리스트로 보내기 위하여 최종 읽을 수를 각 번지별로 저장해야 하며, 또한 각 번지별로 지금까지 읽은 수를 기록하고 있어서 한 번 읽을 때마다 그 값을 증가시키고 최종 카운터와 비교하여 다 읽었는지를 판단하여야 한다.

<17> 이 메모리 영역의 읽기는 매 클럭 일어 날 수 있는데, 이 읽기-래치-증가-쓰기 행위는 한 클럭에 이루어질 수 없으며 메모리의 액세스 타임을 고려하여 일단 래치한 후에 사용하여야 하므로 읽기-래치-증가-쓰기의 순서로 네 클럭에 한 번 이루어져야 한다.

이때, 동일한 메모리 번지의 액세스가 연속해서 이루어질 수 있으므로 어떤 경우에는 앞에서 읽은 데이터의 증가된 값이 쓰여지기 전에 그 번지를 읽을 수도 있어서 틀린 값을 읽게 될 수 있고, 또한 동일한 번지에 동시에 읽기와 쓰기를 할 수 없다는 제약이 있는 경우에도 문제가 발생한다.

<18> 따라서, 이와 같이 연속으로 메모리 내부의 카운터를 읽고 수정하여 쓰는 경우에 메모리의 값의 일관성을 유지할 수 있는 방안이 필수적으로 요구된다.

【발명이 이루고자 하는 기술적 과제】

<19> 본 발명은, 상기한 바와 같은 요구에 부응하기 위하여 제안된 것으로, 공유 메모리 스위치 등에서 최대의 속도로 포인터(데이터가 저장된 주소)의 멀티캐스팅을 위하여 입력 서브큐의 포인터 내용을 해당하는 여러 개의 출력 서브큐로 옮기는 멀티캐스팅 장치 및 그 방법과 상기 방법을 실현시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체를 제공하는데 그 목적이 있다.

<20> 또한, 연속으로 메모리 내부의 카운터를 읽고 수정하여 쓰는 경우에 메모리 값의 일관성을 유지하는데 다른 목적이 있다.

【발명의 구성 및 작용】

<21> 상기 목적을 달성하기 위한 본 발명은, 공유 메모리 기반 스위치에서의 멀티캐스팅 장치에 있어서, 데이터를 읽어 출력 포트 비트 맵에서 한 비트씩 선택하여 한 비트씩의 출력 포트 정보와, 사용하는 경우 클래스 정보를 출력하여 데이터 스트림으로 인에이블

신호와 함께 출력하여 주고, 입력 서브큐에 데이터가 대기하고 있는 경우 현재 처리하고 있는 데이터가 모두 끝나기 전에 미리 상기 입력 서브큐를 읽어 쉬지 않고 자신의 출력 데이터 스트림을 유지하는 입력 서브큐 읽기 수단; 상기 입력 서브큐 읽기 수단에서 입력되는 비트 맵 형식의 출력 포트 정보를 인코딩하여, 사용하는 경우 클래스 정보를 모아서 출력 서브큐의 큐 번호로 만들어 인에이블 신호와 함께 출력하는 큐번호 인코딩 수단; 및 상기 큐번호 인코딩 수단으로부터 입력되는 정보를 이용하여 상기 출력 서브큐의 비사용 번지를 할당받아 해당 출력 서브큐의 꼬리 번지에 쓰고, 상기 해당 출력 서브큐의 다음에 사용할 꼬리 번지를 새로 받은 비사용 번지로 대체함으로써 해당 포인터를 상기 해당 출력 서브큐에 쓰는 출력 서브큐 쓰기 수단을 포함하여 이루어진 것을 특징으로 한다.

<22> 상기 목적을 달성하기 위한 본 발명은, 공유 메모리 기반 스위치에서의 멀티캐스팅 방법에 있어서, 모든 포인터(데이터가 저장된 주소)를 입력 서브큐에 저장하였다가, 출력 포트 비트 맵과 사용되는 경우 해당 데이터의 클래스 정보에 따라 해당 출력 서브큐로 옮기고, 멀티캐스트 데이터의 경우 동일한 포인터를 다수의 출력 서브큐로 복사하여 옮기되, 상기 포인터를 상기 입력 서브큐에서 상기 해당 출력 서브큐로 옮길 때, 상기 입력 서브큐에서 읽은 데이터에 대해서 한번에 한 비트씩 선택하여 한 비트만 선택된 비트 맵의 스트림으로 만들고, 상기 입력 서브큐에 데이터가 있는 경우 현재 처리하고 있는 데이터가 모두 처리되기 전에 미리 상기 입력 서브큐의 데이터를 읽음으로써 상기 입력 서브큐의 데이터와 데이터 사이에 대기 시간이 없도록 하는 것을 특징으로 한다.

<23> 그리고, 본 발명은 공유 메모리 기반 스위치에서의 멀티캐스팅 방법에 있어서, 각 공유 메모리의 저장장소마다 각 번지가 멀티캐스팅을 위해 읽혀져야 하는 최종 숫자가

제1 소정의 메모리에 각 공유메모리 번지마다 저장되어 있고, 각 공유메모리 번지마다 각 시점에서 그 때까지 해당 번지를 읽은 숫자가 상기 제1 소정의 메모리와는 다른 제2 소정의 메모리에 저장되어 있어서 매번 해당 공유 메모리 번지를 읽을 때마다 그 읽은 숫자가 증가되며, 그 값이 최종 숫자와 비교되어 그 값이 최종값보다 작을 경우 일('1') 씩 증가되고, 두 값이 같을 경우 영('0')으로 되어 쓰여지면서 해당 공유 메모리 번지가 비사용 번지 리스트로 반환되는 것을 특징으로 한다.

<24> 상기 목적을 달성하기 위한 본 발명은, 프로세서를 구비한 멀티캐스팅 장치에, 모든 포인터(데이터가 저장된 주소)를 입력 서브큐에 저장하였다가, 출력 포트 비트 맵과 사용되는 경우 해당 데이터의 클래스 정보에 따라 해당 출력 서브큐로 옮기고, 멀티캐스트 데이터의 경우 동일한 포인터를 다수의 출력 서브큐로 복사하여 옮기되, 상기 포인터를 상기 입력 서브큐에서 상기 해당 출력 서브큐로 옮길 때, 상기 입력 서브큐에서 읽은 데이터에 대해서 한번에 한 비트씩 선택하여 한 비트만 선택된 비트 맵의 스트림으로 만들고, 상기 입력 서브큐에 데이터가 있는 경우 현재 처리하고 있는 데이터가 모두 처리되기 전에 미리 상기 입력 서브큐의 데이터를 읽음으로써 상기 입력 서브큐의 데이터와 데이터 사이에 대기 시간이 없도록 하는 기능을 실현시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체를 제공한다.

<25> 그리고, 본 발명은 프로세서를 구비한 멀티캐스팅 장치에, 각 공유 메모리의 저장 장소마다 각 번지가 멀티캐스팅을 위해 읽혀져야 하는 최종 숫자가 제1 소정의 메모리에 각 공유메모리 번지마다 저장되어 있고, 각 공유메모리 번지마다 각 시점에서 그 때까지 해당 번지를 읽은 숫자가 상기 제1 소정의 메모리와는 다른 제2 소정의 메모리에 저장되어 있어서 매번 해당 공유 메모리 번지를 읽을 때마다 그 읽은 숫자가 증가되며, 그

값이 최종 숫자와 비교되어 그 값이 최종값보다 작을 경우 일('1')씩 증가되고, 두 값이 같을 경우 영('0')으로 되어 쓰여지면서 해당 공유 메모리 번지가 비사용 번지 리스트로 반환되는 기능을 실현시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체를 제공한다.

<26> 본 발명에서는 크게, 첫째로 최대의 속도로 포인터의 멀티캐스팅을 하기 위하여 입력 서브큐의 포인터 내용을 해당하는 여러 개의 출력 서브큐로 옮기는 방법과, 둘째로 출력 포트-클래스 서브 큐를 서비스하는 과정에서 멀티캐스팅을 위해서 동일한 번지를 정해진 수 만큼 읽은 후에 비사용 리스트로 반환하기 위해 메모리내의 카운터를 4클럭에 걸쳐 읽어 최종값과 비교하여 증가시키거나 0으로 만들어 다시 쓰는 작용을 매 클럭 실시할 경우 발생하는 문제(즉, 아직 쓰지 않은 값을 뒤에서 읽거나 읽기와 쓰기를 동일한 어드레스에 대해 할 수 없음으로 인해 쓰지 못한 데이터를 뒤에서 읽는 경우에 발생하는 일관성(coherency) 문제)를 해결하는 것이다.

<27> 입력 서브큐에서 출력 서브큐로 포인터를 옮기는 과정은 입력 서브큐에 옮길 포인터가 있을 경우 라우팅 정보에 따라 여러 개의 출력 서브큐로 옮기는 것이다. 이때, 라우팅 정보는 비트 맵의 형태를 가지고 있으며, 비트 맵과 함께 가지고 있는 해당 클래스 정보를 함께 고려하여 해당하는 출력 서브큐로 옮긴다. 이때, 입력 서브큐에 읽을 데이터가 있을 경우에 앞의 데이터를 처리하고 나서 다음 데이터를 처리하는 중간에는 출력 서브큐를 쓰는 행위가 중단되지 않도록 입력 서브큐를 미리 읽어야 한다.

<28> 또한, 공유 메모리의 특정 번지에 있는 데이터가 읽혀져야 할 수는 다른 메모리의 동일한 번지에 미리 기록되고 각 번지의 데이터가 읽혀진 수는 별도의 메모리에 각 번지별로 기록되어 있는데 해당 번지의 데이터가 읽혀질 때마다 그 카운터를 읽어 최종 값과

비교하여 최종 값보다 작으면 1씩 증가하고, 최종 값과 같으면 0으로 만들어 다시 쓰면서 동시에 해당 공유 메모리 번지를 비사용 리스트로 반환한다.

<29> 한 번지에 대한 이러한 읽기-수정-쓰기는 한 클럭에 이루어질 수 없으므로 제일 빠르게 구현할 경우 읽기-래치-변환-쓰기로 나뉘어 질 수 있다. 이러한 동작이 각 번지에 대해서 매 클럭 일어나고 동시에 같은 번지에 읽고 쓰기가 불가능하므로 앞에서 언급한 바와 같은 동일성 문제가 발생한다. 이러한 '쓰기 전에 읽는' 문제와 '쓰지 못한 데이터를 읽는' 문제는 읽은 데이터를 무효로 하고 나중에 쓰게 된 데이터를 사용하거나 쓰지 못한 데이터를 나중에 사용함으로써 해결할 수 있다.

<30> 결론적으로, 본 발명은 공유 메모리 방식의 스위치에서 포인터가 일단 입력 서브큐에 저장되었다가 나중에 출력 포트 비트 맵에 따라 해당하는 출력 서브큐로 움직이도록 하고, 멀티캐스팅의 경우 동일한 값이 여러 개의 출력 서브큐로 복사되도록 하는 방식을 사용하는 경우에 입력 서브큐의 상태, 즉 처리할 데이터가 있는지의 여부와 그 데이터가 마지막 데이터인지의 여부를 이용하고, 입력 서브큐에서 읽어낸 데이터, 읽어낸 후에 처리되지는 않았지만 기다리고 있는 데이터, 처리중에 있으면서 아직 선택되지 않은 포트의 비트들이 모아져 있는 데이터, 선택된 포트 비트를 가지고 있는 데이터를 각각 값으로 갖는 레지스터들과 각 레지스터들이 데이터를 갖고 있는지의 여부, 어떤 데이터를 갖고 있는지의 여부를 이용하여, 입력 서브큐에서 한 데이터씩 읽어 한 비트씩 선택해서 출력 서브큐로 옮기는 과정에서 한 클럭도 잃어버리지 않고 이러한 과정을 수행할 수 있다.

<31> 또한, 본 발명은 각 공유 메모리 번지가 읽혀져야 할 최종 숫자가 별도의 메모리에 번지별로 보관되고, 각 공유메모리 번지가 읽혀진 숫자를 기록하는 카운터를 다른 별도

의 메모리에 보관하면서 각 번지를 읽을 때마다 해당 번지의 읽기 카운터를 증가시켜 최종 카운터와 비교함으로써 해당 번지를 비사용 공유 메모리 번지 리스트로 반환시켜야 할지 아닌지를 알아내는 경우에, 읽기, 래치, 수정, 쓰기의 순서로 네 클럭에 걸쳐 처리하되, 매 클럭 새로운 번지를 처리하고 동일한 번지에 대해서는 읽기와 쓰기가 이루어질 수 없는 경우에는 쓰기를 막도록 하면서 이러한 이유로 쓰여지지 못한 데이터와 처리지연으로 아직 쓰여지지 않은 데이터가 있는 경우 이 조건을 검출하여 실제 읽은 데이터 대신 사용하게 함으로써 카운터의 증가 및 반환조건을 정확하게 검출한다.

<32> 상술한 목적, 특징들 및 장점은 첨부된 도면과 관련한 다음의 상세한 설명을 통하여 보다 분명해 질 것이다. 이하, 첨부된 도면을 참조하여 본 발명에 따른 바람직한 일 실시예를 상세히 설명한다.

<33> 먼저, 포인터를 입력 서브큐에서 출력 서브큐로 옮기는 방법에 대해 보다 상세히 설명한다.

<34> 본 발명이 적용되는 공유 메모리 스위치에서 사용되는 큐의 구조는 도 1과 같이 나타낼 수 있다.

<35> 도 1에 도시된 바와 같이, 모든 입력 포트에서 들어온 데이터는 유효한 데이터에 대해서 공유 메모리의 번지를 할당받고, 이 공유 메모리 번지와 해당 데이터가 향하고 있는 출력 포트의 비트맵 값, 해당 데이터의 클래스 값이 한 단위가 되어 순차적으로 입력 서브큐에 옮겨진 후, 입력 서브큐에 있는 데이터들이 하나씩 읽혀져서 해당하는 출력 포트 및 클래스에 따라 해당 출력 서브큐로 옮겨진다.

<36> 입력 포트의 수가 p, 클래스의 수가 c라고 가정하면, 전체 서브큐의 갯수는 'p x c'

가 된다.

<37> 포인터를 입력 서브큐에서 출력 서브큐로 옮기는 멀티캐스팅 장치는, 크게 세 부분으로 나뉘어 진다. 이를 도 2를 통해 보다 상세히 설명한다.

<38> 도 2에서, 'en1', 'cssba1', 'pid1', 'pri1'은 각각 인에이블 신호, 패킷이 저장된 공유 메모리 번지, 비트 맵 형식으로 바뀌고 한 비트만 1로 셋트되어 있는 포트 식별자, 그리고 해당 패킷의 우선순위(클래스)이다.

<39> 그리고, 'en2', 'cssba2', 'pid2', 'pri2'는 각각 큐번호 변환부(20)의 출력에서의 인에이블, 공유 메모리 번지, 인코딩된 형태의 출력 포트 식별자, 그리고 우선순위를 나타낸다.

<40> 도 2에 도시된 바와 같이, 포인터를 입력 서브큐에서 출력 서브큐로 옮기는 멀티캐스팅 장치는, 데이터를 읽어 출력 포트 비트 맵에서 한 비트씩 선택하여 한 비트씩의 출력 포트 정보와, 사용하는 경우 클래스 정보를 출력하여 데이터 스트림으로 인에이블 신호와 함께 출력하여 주고, 입력 서브큐에 데이터가 대기하고 있는 경우 현재 처리하고 있는 데이터가 모두 끝나기 전에 미리 입력 서브큐를 읽어 쉬지 않고 자신의 출력 데이터 스트림을 유지하는 입력큐 읽기부(10)와, 입력큐 읽기부(10)에서 입력되는 비트 맵 형식의 출력 포트 정보를 인코딩하여, 사용하는 경우 클래스 정보를 모아서 출력 서브큐의 큐 번호로 만들어 인에이블 신호와 함께 출력하는 큐번호 변환부(20)와, 큐번호 변환부(10)로부터 입력되는 정보를 이용하여 출력 서브큐의 비사용 번지를 할당받아 해당하는 출력 서브큐의 꼬리 번지에 쓰고, 해당 출력 서브큐의 다음에 사용할 꼬리 번지를 새로 받은 비사용 번지로 대체함으로써 해당 포인터를 해당 출력 서브큐에 쓰는 출력큐 쓰기부(30)를 포함한다.

<41> 여기서, 모든 포인터는 일단 입력 서브큐에 저장되었다가 출력 포트 비트 맵과 사용되는 경우 해당 데이터의 클래스 정보에 따라 해당 출력 서브큐로 옮겨지고, 멀티캐스트 데이터의 경우 동일한 포인터가 여러 개의 출력 서브큐로 복사되어 옮겨진다.

<42> 그리고, 입력 서브큐에서 읽은 데이터에 대해서 한번에 한 비트씩 선택하여 한 비트만 선택된 비트 맵의 스트림으로 만들고, 입력 서브 큐에 데이터가 있는 경우 현재 처리하고 있는 데이터가 모두 처리되기 전에 미리 입력 서브큐의 데이터를 읽음으로써 입력 서브큐의 데이터와 데이터 사이에 대기 시간이 없도록 한다.

<43> 입력큐 읽기부(10)는 입력 서브큐에서 출력 서브큐로 포인터를 이동시키는 처리가 입력 서브큐를 읽어 인에이블 신호와 한 비트만 1로 셋트된 비트 맵 형태의 포트 맵, 그리고 사용할 경우에 클래스 정보를 스트림으로 출력해 준다.

<44> 큐번호 변환기(20)는 입력큐 읽기부(10)로부터 입력된 정보를 받아 인에이블 신호와 인코딩된 포트 정보를 스트림으로 출력해 준다.

<45> 출력큐 쓰기부(30)는 큐번호 변환기(20)로부터 정보를 받아 해당하는 출력 서브큐에 대해서 비사용 출력 서브큐 번지를 받아 포인터와 함께 해당 출력 서브큐에 씀으로써 포인터를 출력 서브큐에 쓰면서 링크드 리스트를 연결하고 동시에 사용한 비사용 출력 서브큐 번지를 해당 출력 서브큐의 다음에 사용할 꼬리 번지로서 보관하는 일을 담당한다.

<46> 입력큐 읽기부(10)는 입력 서브큐에 데이터가 있는지, 또한 그것이 유일한 데이터인지 아닌지의 정보를 받는데 이 정보, 즉 유일한 데이터인지 아닌지의 정보는 입력 서브큐 읽기 신호가 입력 서브큐의 데이터가 있는지 없는지를 나타내는 신호가 업데이트되

기 전에 미리 출력되어야 하므로 멈출지 진행할지를 미리 알아내기 위해 필요하다.

<47> 그리고, 입력큐 읽기부(10)는 주 제어 블럭으로부터 동작 인에이블 신호를 받게 되며, 또한 출력큐 쓰기부(30)로부터 준비되었는지의 신호를 받아 준비가 된 상황에서 동작을 하게 된다. 여기서, 출력 서브큐는 링크드 리스트로 구성되는 경우에는 출력큐 쓰기부(30)가 초기에 각 출력 서브큐별로 비사용 번지를 최초로 사용할 꼬리 번지로서 가지고 있어야 하므로 초기에 비사용 번지를 가져오는 시간이 필요하게 되고 준비되었는지를 나타내는 신호가 필요한 것이다.

<48> 입력 서브큐의 읽기의 결과를 서브셀이라고 하면 이 서브셀 데이터는 목적지 포트의 비트 맵과 클래스, 그리고 공유 메모리에 데이터가 저장된 번지를 가지고 있게 된다.

<49> 출력큐 쓰기부(30)는 포인터 값을 큐 식별자에 따라 해당하는 출력 서브큐에 써 넣는 작용을 하는데, 포인터를 현재의 출력 서브큐 꼬리 번지에 써 넣고, 동시에 비사용 출력 서브큐 번지를 가져와 해당 출력 서브큐의 다음 꼬리 번지로 사용한다. 이 새 꼬리 번지는 포인터를 써 넣을 때 같이 써 넣음으로써 출력 서브큐 내에서 링크드 리스트로 패킷들이 연결될 수 있도록 한다. 출력 서브큐의 비사용 번지를 가져 오는데는 시간이 걸리므로, 출력큐 쓰기부(30) 내부에는 파이프 라인 방법을 사용하여 새 비사용 번지를 가져오는 동안 입력된 데이터가 지연되었다가 사용되도록 해야 한다.

<50> 그리고, 출력큐 쓰기부(30)는 출력 서브큐의 증가 정보를 외부의 큐 레벨 카운팅 블럭, 또는 미터 블럭으로 보내어 스케쥴링이 이루어질 수 있도록 한다.

<51> 입력 서브큐에서 출력 서브큐로 포인터를 옮기는 데 있어서의 성능은 입력큐 읽기

부(10)가 좌우하게 된다. 이러한 입력큐 읽기부(10)의 내부 구성이 도 3에 도시되었다.

<52> 도 3에 도시된 바와 같이, 입력큐 읽기부(10)는 데이터의 이동 및 선택을 제어하기 제어부(18)와, 입력 서브큐를 읽은 데이터(서브셀)를 래치하여 보관하고 있는 읽은값 레지스터(11)와, 대기값 레지스터(13)의 다음값(대기값)을 계산하기 위한 대기값 계산부(대기값을 계산하기 위한 조합논리회로)(12)와, 제어부(18)의 제어 명령(래치 인에이블 신호와 데이터 선택신호)에 따라, 현재 처리되지 않고 대기중인 서브셀의 값을 가지고 있는 대기값 레지스터(13)와, 남은값 레지스터(15)의 다음값(남은값)을 계산하기 위한 남은값 계산부(남은값을 계산하기 위한 조합논리회로)(14)와, 제어부(18)의 제어 명령(래치 인에이블 신호와 데이터 선택신호)에 따라, 현재 처리되고 있는 데이터에 해당하면서 아직 선택되지 않은 비트들을 가지고 있는 남은값 레지스터(15)와, 선택값 레지스터(17)의 다음값(선택값)을 계산하기 위한 선택값 계산부(선택값을 계산하기 위한 조합논리회로)(16)와, 제어부(18)의 제어 명령(래치 인에이블 신호와 데이터 선택신호)에 따라, 한 비트씩 선택되어 래치된 선택값 레지스터(17)를 포함한다.

<53> 제어부(18)는 입력큐 읽기부(10)의 주요 입출력을 담당할 뿐만 아니라, 블럭 내부의 각 대기값 레지스터(13), 남은값 레지스터(15), 선택값 레지스터(15)의 각 래치 인에이블 신호와 데이터 선택신호를 발생하는 역할을 한다.

<54> 데이터를 각 레지스터 단에서 다음 레지스터 단으로 옮기기 위한 조건은 각 단의 상태와 다음 단위 상태, 때로는 이전 단위 상태에서 독립적으로 유도할 수 있다. 또한, 입력 서브큐 읽기 신호를 어서트할 조건 등도 동작 인에이블신호, 출력큐 쓰기부(30)의 준비신호, 입력 서브큐의 상태, 읽은값(서브셀) 레지스터(11)에 데이터가 있는지 없는지, 대기값 레지스터(13)에 데이터가 있는지 없는지, 남은값

레지스터(15)에 남은 데이터가 있는지 없는지에 의해 정해진다.

<55> 입력 서브큐 읽기 신호를 발생하는 조건, 각 레지스터의 값을 선택하는 조건이 도 4a 내지 4e에 도시되었다. 도 4a 내지 4e에서 각 레지스터에 값이 존재하는 지는 해당 레지스터의 플래그를 보고 알게 되는데, 각 레지스터 플래그의 설정조건도 표시하였다. 이러한 과정은 매 클럭마다 반복 수행된다.

<56> 이를 구체적으로 살펴보면 다음과 같다.

<57> 도 4a를 참조하여 입력 큐 읽기 어서트 과정을 살펴보면, 만약 동작지시가 있고 출력큐 쓰기가 준비상태에서(401), 입력큐가 비어 있지 않고 현재 마지막 하나를 읽고 있는 중이 아니고(402), 읽은값 레지스터에 아무값도 없고 대기값 레지스터에 아무값도 없으며 읽은값 레지스터에 데이터가 들어가기 직전이 아니면(403), 입력큐 읽기신호를 어서트한다(404).

<58> 한편, 동작지시가 있고 출력큐 쓰기가 준비상태에서(401), 입력큐가 비어 있지 않고 현재 마지막 하나를 읽고 있는 중이 아니고(402), 읽은값 레지스터에 아무값도 없고 대기값 레지스터에 아무값도 없으며 읽은값 레지스터에 데이터가 들어가기 직전이며(403), 대기값 레지스터에 데이터가 있고 대기값 레지스터에 한 비트만 남아 있고 남은값 레지스터에 아무것도 없으면(405), 입력큐 읽기신호를 어서트한다(404).

<59> 다른 한편, 동작지시가 있고 출력큐 쓰기가 준비상태에서(401), 입력큐가 비어 있지 않고 현재 마지막 하나를 읽고 있는 중이 아니고(402), 읽은값 레지스터에 아무값도 없고 대기값 레지스터에 아무값도 없으며 읽은값 레지스터에 데이터가 들

어가기 직전이며(403), 대기값 레지스터에 데이터가 있고 대기값 레지스터에 한 비트만 남아 있고 남은값 레지스터에 남아 있는 값이 있으며(405), 읽은값 레지스터에 데이터가 있고 대기값 레지스터도 데이터가 있고 남은값 레지스터에 한 비트만 남아 있으면(406), 입력큐 읽기신호를 어서트한다(404).

<60> 도 4b를 참조하여 읽은값 레지스터 값 및 그 플래그 셋팅 과정을 살펴보면, 만약 한 클럭전에 입력큐를 읽었으면(411) 읽은값 레지스터 플래그를 1로 셋팅하고 입력큐의 출력값을 읽은값 레지스터에 입력한다(412).

<61> 한편, 한 클럭전에 입력큐를 읽지 않고(411), 읽은값 레지스터에 데이터가 있으면서 대기값 레지스터에 데이터가 없고 남은값 레지스터에 데이터가 없는 조건이면(413) 읽은값 레지스터의 플래그를 0으로 리셋한다(414).

<62> 도 4c를 참조하여 대기값 레지스터 값 및 그 플래그 셋팅 과정을 살펴보면, 만약 읽은값 레지스터에 데이터가 있으면서 대기값 레지스터에 데이터가 없거나 남은값 레지스터에 아무값도 없는 조건이면(421), 대기값 레지스터의 플래그를 1로 셋팅하고 입력큐의 출력값을 대기값 레지스터에 입력한다(422).

<63> 한편, 읽은값 레지스터에 데이터가 없으면서(421) 대기값 레지스터에 데이터가 있고 남은값 레지스터에 아무값도 없으면(423), 대기값 레지스터의 플래그를 0으로 리셋한다(424).

<64> 도 4d를 참조하여 남은값 레지스터 값 결정 과정을 살펴보면, 남은값 레지스터에 데이터가 없으면(431), 다음 클럭 주기의 남은값 레지스터는 남은값 레지스터에서 최상위 비트를 제외한 값이 된다(435).

<65> 한편, 남은값 레지스터에 데이터가 있고(431), 대기값 레지스터에 데이터가 있으면(432), 다음 클럭 주기의 남은값 레지스터는 대기값 레지스터에서 최상위 비트를 제외한 값이 된다(433).

<66> 다른 한편, 남은값 레지스터에 데이터가 있고(431), 대기값 레지스터에 데이터가 없으면(432), 다음 클럭 주기의 남은값 레지스터는 0이 된다(434).

<67> 도 4e를 참조하여 선택값 레지스터 값 결정 과정을 살펴보면, 만약 남은값 레지스터에 데이터가 없으면(441), 다음 클럭 주기의 선택값 레지스터는 남은값 레지스터에서 최상위 비트만 선택한 값이 된다(445).

<68> 한편, 만약 남은값 레지스터에 데이터가 있으며(441), 대기값 레지스터에 데이터가 있으면(442), 다음 클럭 주기의 선택값 레지스터는 대기값 레지스터에서 최상위 비트만 선택한 값이 된다(443).

<69> 다른 한편, 만약 남은값 레지스터에 데이터가 있으며(441), 대기값 레지스터에 데이터가 없으면(442), 다음 클럭 주기의 선택한 레지스터는 0이 된다(444).

<70> 이제, 메모리 내부의 읽기 카운터 관리 방안에 대해 보다 상세하게 설명한다.

<71> 본 실시예에서 메모리에의 쓰기는 쓰기 번지와 쓰기 데이터를 쓰기 인에이블 신호와 함께 같은 클럭 주기에 제공함으로써 이루어진다고 가정한다.

<72> 메모리로부터의 읽기는 읽기 번지와 읽기 인에이블 신호를 제공하면, 어떤 클럭 주기에 제공하는 경우에 다음 클럭 주기에 데이터가 출력되는 동기식 메모리로 가정한다. 만약, 비동기식 메모리를 사용할 경우에는 데이터가 읽기 번지를 제공하는 같은 클럭 주기에 출력되므로 한 번 래치한 데이터는 동기식 메모리와 같은 타이밍을 가지게 되며 클

럭의 시작점부터 깨끗한 데이터를 얻을 수 있으나, 대신에 읽기 번지를 제공하는 것을 빨리 할 수 있어야 하므로 장단점이 있다. 읽기의 경우 읽기 인에이블 신호는 항상 인에이블된 상태로 고정시킬 수 있다.

<73> 도 5 는 본 발명에 따른 멀티캐스트를 위해 메모리내의 카운터를 읽기 및 쓰기하는 과정을 나타낸 일실시예 설명도이다.

<74> 클럭 주기 N에서 읽기 번지 RD_ADDR(N)이 주어지고 읽기 데이터는 클럭 주기 N+1에 오게 된다. 동기식 메모리는 액세스 타임이 필요하므로 메모리의 출력은 클럭 시작점에 비해 동작 클럭이 높을 경우 상당한 양의 시간지연을 갖게 된다. 따라서, 메모리의 출력 데이터를 곧바로 사용하는 것은 힘들며 한 번 래치하여 주기 N+2에서 사용하는 것이 보통이다. 읽어낸 데이터가 최종값과 같은지, 따라서 그 값을 1만큼 증가시켜야 할 지, 아니면 0으로 만들어야 할 지도 클럭 주기 N+2에서 이루어진다. 최종 결정된 값은 클럭 주기 N+3에서 메모리에 다시 쓰여지게 된다.

<75> 여기서는 블럭킹(blocking)이라는 말을 정의한다. 대부분의 동기식 이중포트 메모리의 요구사항에 의해 읽기와 쓰기의 번지가 같을 경우 읽기의 데이터가 제대로 나올 수 없게 된다. 만약, 쓰기할 번지가 공교롭게도 현재 읽고 있는 번지와 같을 경우에는 조합논리를 사용하여 메모리의 쓰기를 막아 주어야 한다. 블럭킹에 의해 쓰기가 막혔을 때에는 이 쓰려고 했던 값은 뒤의 처리, 즉 블럭킹을 유발시켰던 읽기 행위를 포함한 뒤의 처리로 전달되어 실제 읽은 데이터 대신에 더 적법한 값으로서 사용되어야 한다. 실제 데이터가 쓰여지게 되면 그 후에는 실제로 읽혀진 데이터가 사용되면 된다.

<76> 클럭 주기 N에서 어떤 특정 번지를 읽고 있었다고 가정하자.

<77> 만약, 주기 N에서 우연히 같은 번지에 쓰기를 하려다가 블럭킹이 일어났다면 원래 주기 N에서 쓰려고 했던 값은 클럭 주기 N+2까지 전달되어 마치 주기 N에서 읽은 값처럼 여겨져서 주기 N+3에서 쓸 값을 계산하는데 사용되어야 한다. 이 경우에 실제로 주기 N에서 읽은 값은 무시된다.

<78> 만약, 클럭 주기 N-1에서 같은 번지에 블럭킹이 일어났다면, 그리고 그 블럭킹 번지가 클럭 주기 N에서 읽고 있는 번지와 같았다(이러한 조건은 클럭 주기 N-1에서는 알 수 없고 주기 N이 되어서야 알 수 있으므로 이러한 조건을 검출하기 위해서는 읽기 번지와 블럭킹이 있었는지의 여부가 과거 사실로서 지연되면서 보관되어야 함)고 하면, 그리고 현재의 클럭 주기 N에서는 블럭킹이 없다고 하면, 클럭 주기 N-1에서 원래 쓰려고 했던 값은 클럭 주기 N+2까지 전달되어 마찬가지로 마치 주기 N에서 읽은값처럼 사용되어져야 한다. 이 경우 역시 클럭 주기 N에서 읽은 값은 무시되어야 한다.

<79> 한편, 클럭 주기 N-2에서 블럭킹이 있었고 그 번지가 현재의 주기 N에서 읽고 있는 번지와 같았다면(이 조건 역시 주기 N-2에서는 모르고 주기 N이 되어서야 알 수 있으므로 읽기 번지와 블럭킹이 있었는지의 여부가 2번 지연되어 사용되어져야 함), 그리고 주기 N-1에 같은 번지에 블럭킹이 없었고 주기 N에도 현재 블럭킹이 없다면, 클럭 주기 N-2에서 원래 쓰려고 했던 값은 클럭 주기 N+2까지 전달되어 마찬가지로 마치 클럭 주기 N에서 읽은 값처럼 사용되어져야 한다. 이 경우 역시 주기 N에서 읽은 데이터는 무시된다.

<80> 이렇게 데이터를 무시하는 것은 읽은 데이터가 지연되는 과정에서 0으로 리셋되게 함으로써도 가능하고 원하는 데이터를 선택함으로써도 가능하다. 또는, 두 가지를 조합할 수도 있다.

<81> 위의 사항들은 앞의 처리한 데이터가 블럭킹에 의해 쓰여지지 않았을 경우를 대비한 것들이었다. 그러나, 클럭 주기 N에서의 읽기가 클럭 주기 N이전에 쓰여진 최종 쓰기 데이터를 읽은 경우라고 하더라도 클럭 주기 N+1이나 클럭 주기 N+2 에서 같은 번지에 쓰기가 일어날 수 있다. 이는 클럭 주기 N-2나 클럭 주기 N-1에서 주기 N에서 읽고 있는 번지를 읽었을 때 벌어질 수 있는 상황이다. 이러한 경우에는 주기 N+1이나 N+2에서 쓰여진 데이터가 마치 주기 N에서 읽은값처럼 사용되어져야 한다. 이 문제는 클럭 주기 N-2와 클럭 주기 N-1에서 현재의 클럭 주기 N에서 읽고 있는 동일한 번지를 읽은 수를 세어서 그 값을 실제 주기 N에서 읽은 값에 더해 줌으로써 보상해 줄 수 있다.

<82> 상술한 바와 같은 본 발명의 방법은 프로그램으로 구현되어 컴퓨터로 읽을 수 있는 기록매체(씨디롬, 램, 롬, 플로피 디스크, 하드 디스크, 광자기 디스크 등)에 저장될 수 있다.

<83> 이상에서 설명한 본 발명은 전술한 실시예 및 첨부된 도면에 의해 한정되는 것이 아니고, 본 발명의 기술적 사상을 벗어나지 않는 범위 내에서 여러 가지 치환, 변형 및 변경이 가능하다는 것이 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 있어 명백할 것이다.

【발명의 효과】

<84> 상기한 바와 같은 본 발명은, 입력 서브큐의 포인터 내용을 여러 개의 출력 서브큐로 최대의 속도로 멀티캐스팅할 수 있고, 이때 한 클럭도 잃어 버리지 않고 매번 출력 서브큐로 포인터를 옮길 수 있으며, 연속으로 메모리 내부의 카운터를 읽고 수정하여 쓰

는 경우에 메모리 값의 일관성을 유지시킬 수 있는 효과가 있다.

【특허청구범위】

【청구항 1】

공유 메모리 기반 스위치에서의 멀티캐스팅 장치에 있어서,

데이터를 읽어 출력 포트 비트 맵에서 한 비트씩 선택하여 한 비트씩의 출력 포트 정보와, 사용하는 경우 클래스 정보를 출력하여 데이터 스트림으로 인에이블 신호와 함께 출력하여 주고, 입력 서브큐에 데이터가 대기하고 있는 경우 현재 처리하고 있는 데이터가 모두 끝나기 전에 미리 상기 입력 서브큐를 읽어 쉬지 않고 자신의 출력 데이터 스트림을 유지하는 입력 서브큐 읽기 수단;

상기 입력 서브큐 읽기 수단에서 입력되는 비트 맵 형식의 출력 포트 정보를 인코딩하여, 사용하는 경우 클래스 정보를 모아서 출력 서브큐의 큐 번호로 만들어 인에이블 신호와 함께 출력하는 큐번호 인코딩 수단; 및

상기 큐번호 인코딩 수단으로부터 입력되는 정보를 이용하여 상기 출력 서브큐의 비사용 번지를 할당받아 해당 출력 서브큐의 꼬리 번지에 쓰고, 상기 해당 출력 서브큐의 다음에 사용할 꼬리 번지를 새로 받은 비사용 번지로 대체함으로써 해당 포인터를 상기 해당 출력 서브큐에 쓰는 출력 서브큐 쓰기 수단

을 포함하는 공유메모리 스위치에서의 멀티캐스팅 장치.

【청구항 2】

제 1 항에 있어서,

상기 입력 서브큐 읽기 수단은,

데이터의 이동 및 선택을 제어하기 제어수단;

상기 입력 서브큐를 읽은 데이터(서브셀)를 래치하여 보관하고 있는 서브셀 저장수단;

대기값 저장수단의 다음값(대기값)을 계산하기 위한 대기값 계산수단;

상기 제어수단의 제어 명령(래치 인에이블 신호와 데이터 선택신호)에 따라, 현재 처리되지 않고 대기중인 서브셀의 값을 가지고 있는 상기 대기값 저장수단;

남은값 저장수단의 다음값(남은값)을 계산하기 위한 남은값 계산수단;

상기 제어수단의 제어 명령(래치 인에이블 신호와 데이터 선택신호)에 따라, 현재 처리되고 있는 데이터에 해당하면서 아직 선택되지 않은 비트들을 가지고 있는 상기 남은값 저장수단;

선택값 저장수단의 다음값(선택값)을 계산하기 위한 선택값 계산수단; 및

상기 제어수단의 제어 명령(래치 인에이블 신호와 데이터 선택신호)에 따라, 한 비트씩 선택되어 래치된 상기 선택값 저장수단

을 포함하는 공유메모리 스위치에서의 멀티캐스팅 장치.

【청구항 3】

제 1 항 또는 제 2 항에 있어서,

상기 포인터(데이터가 저장된 주소)는,

우선 상기 입력 서브큐에 저장되었다가 출력 포트 비트 맵과 사용되는 경우 해당 데이터의 클래스 정보에 따라 상기 해당 출력 서브큐로 옮겨지고, 멀티캐스트

데이터의 경우 동일한 포인터가 여러 개의 출력 서브큐로 복사되어 옮겨지는 것을 특징으로 하는 공유메모리 스위치에서의 멀티캐스팅 장치.

【청구항 4】

제 3 항에 있어서,

상기 포인터를 상기 입력 서브큐에서 상기 해당 출력 서브큐로 옮길 때,

상기 입력 서브큐에서 읽은 데이터에 대해서 한번에 한 비트씩 선택하여 한 비트만 선택된 비트 맵의 스트림으로 만들고 상기 입력 서브큐에 데이터가 있는 경우 현재 처리하고 있는 데이터가 모두 처리되기 전에 미리 상기 입력 서브큐의 데이터를 읽음으로써 상기 입력 서브큐의 데이터와 데이터 사이에 대기 시간이 없도록 하는 것을 특징으로 하는 공유메모리 스위치에서의 멀티캐스팅 장치.

【청구항 5】

공유 메모리 기반 스위치에서의 멀티캐스팅 방법에 있어서;

모든 포인터(데이터가 저장된 주소)를 입력 서브큐에 저장하였다가, 출력 포트 비트 맵과 사용되는 경우 해당 데이터의 클래스 정보에 따라 해당 출력 서브큐로 옮기고, 멀티캐스트 데이터의 경우 동일한 포인터를 다수의 출력 서브큐로 복사하여 옮기되,

상기 포인터를 상기 입력 서브큐에서 상기 해당 출력 서브큐로 옮길 때, 상기 입력 서브큐에서 읽은 데이터에 대해서 한번에 한 비트씩 선택하여 한 비트만 선택된 비트 맵의 스트림으로 만들고, 상기 입력 서브큐에 데이터가 있는 경우 현재 처리하고 있는 데

이터가 모두 처리되기 전에 미리 상기 입력 서브큐의 데이터를 읽음으로써 상기 입력 서브큐의 데이터와 데이터 사이에 대기 시간이 없도록 하는 것을 특징으로 하는 공유메모리 스위치에서의 멀티캐스팅 방법.

【청구항 6】

공유 메모리 기반 스위치에서의 멀티캐스팅 방법에 있어서,

각 공유 메모리의 저장장소마다 각 번지가 멀티캐스팅을 위해 읽혀져야 하는 최종 숫자가 제1 소정의 메모리에 각 공유메모리 번지마다 저장되어 있고, 각 공유메모리 번지마다 각 시점에서 그 때까지 해당 번지를 읽은 숫자가 상기 제1 소정의 메모리와는 다른 제2 소정의 메모리에 저장되어 있어서 매번 해당 공유 메모리 번지를 읽을 때마다 그 읽은 숫자가 증가되며, 그 값이 최종 숫자와 비교되어 그 값이 최종값보다 작을 경우 일('1')씩 증가되고, 두 값이 같을 경우 영('0')으로 되어 쓰여지면서 해당 공유 메모리 번지가 비사용 번지 리스트로 반환되는 것을 특징으로 하는 공유메모리 스위치에서의 멀티캐스팅 방법.

【청구항 7】

제 6 항에 있어서,

상기 읽기 및 쓰기 과정은,

각 공유 메모리의 읽기 번지에 대한 읽기 카운트가 읽기, 래치, 수정, 쓰기

의 순서로 이루어지며, 읽기 번지와 쓰기 번지가 같을 경우에 쓰기가 취소되도록 하고, 블럭킹에 의해 쓰기가 취소된 경우 쓰기 데이터가 뒤로 전달되어 읽은 데이터 대신에 사용되도록 하며, 처리 지연에 의해 아직 쓰여지지 않은 데이터가 있었는데 읽은 경우에 그러한 읽기의 수를 실제 읽은 값에 더함으로써 쓰기가 옳게 일어날 수 있도록 하는 것을 특징으로 하는 공유메모리 스위치에서의 멀티캐스팅 방법.

【청구항 8】

제 7 항에 있어서,

상기 쓰기가 상기 읽기에 대해 동일한 번지에 일어나서 쓰기가 막힌 경우에,

그러한 일이 있었는지와 그 번지를 2클럭까지, 혹은 그 이상 지연시키면서 나중에 이 정보를 이용하여 쓰기를 하지 못한 값을 실제 읽은 값 대신에 사용하는 것을 특징으로 하는 공유메모리 스위치에서의 멀티캐스팅 방법.

【청구항 9】

제 8 항에 있어서,

실제로 읽은 데이터가 사용되어지지 않는 경우에,

지연시키는 도중에 영('0')으로 리셋되어 쓰여지지 않은 값에 더해지거나 사용되어지지 않은 값과 다중화되어 없어지도록 하는 것을 특징으로 하는 공유메모리 스위치에서의 멀티캐스팅 방법.

【청구항 10】

프로세서를 구비한 멀티캐스팅 장치에,

모든 포인터(데이터가 저장된 주소)를 입력 서브큐에 저장하였다가, 출력 포트 비트 맵과 사용되는 경우 해당 데이터의 클래스 정보에 따라 해당 출력 서브큐로 옮기고, 멀티캐스트 데이터의 경우 동일한 포인터를 다수의 출력 서브큐로 복사하여 옮기되,

상기 포인터를 상기 입력 서브큐에서 상기 해당 출력 서브큐로 옮길 때, 상기 입력 서브큐에서 읽은 데이터에 대해서 한번에 한 비트씩 선택하여 한 비트만 선택된 비트 맵의 스트림으로 만들고, 상기 입력 서브큐에 데이터가 있는 경우 현재 처리하고 있는 데이터가 모두 처리되기 전에 미리 상기 입력 서브큐의 데이터를 읽음으로써 상기 입력 서브큐의 데이터와 데이터 사이에 대기 시간이 없도록 하는 기능

을 실현시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체.

【청구항 11】

프로세서를 구비한 멀티캐스팅 장치에,

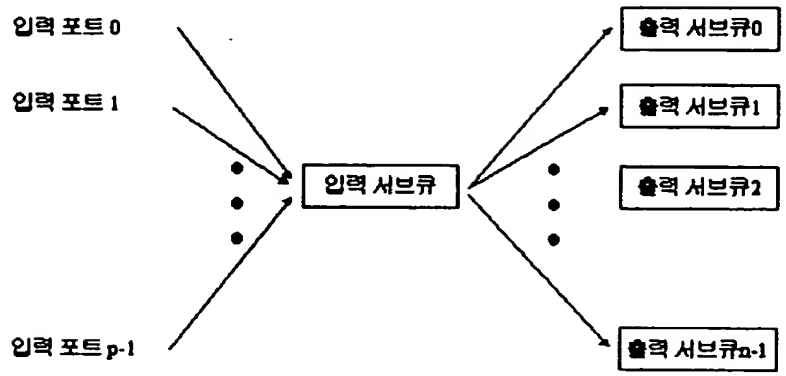
각 공유 메모리의 저장장소마다 각 번지가 멀티캐스팅을 위해 읽혀져야 하는 최종 숫자가 제1 소정의 메모리에 각 공유메모리 번지마다 저장되어 있고, 각 공유메모리 번지마다 각 시점에서 그 때까지 해당 번지를 읽은 숫자가 상기 제1 소정의 메모리와는 다른 제2 소정의 메모리에 저장되어 있어서 매번 해당 공유 메모리 번지를 읽을 때마다 그 읽은 숫자가 증가되며, 그 값이 최종 숫자와 비교되어 그 값이 최종값보다 작을 경우 일('1')씩 증가되고, 두 값이 같을 경우 영('0')으로 되어 쓰여지면서 해당 공유 메모리 번

지가 비사용 번지 리스트로 반환되는 기능

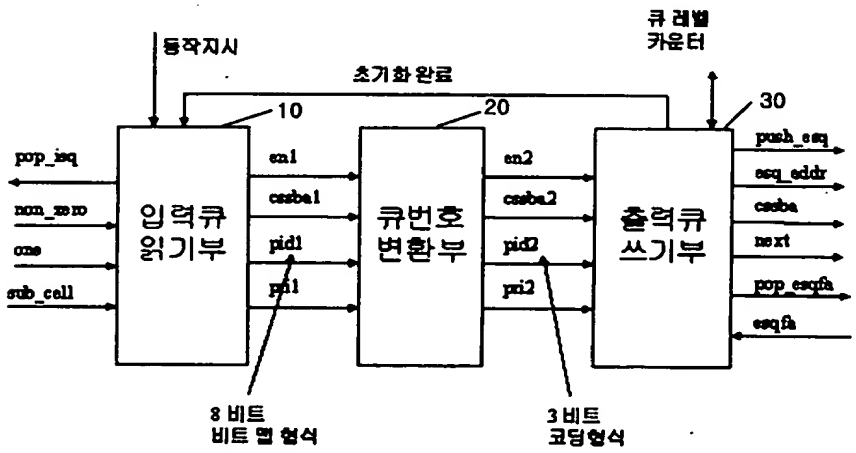
을 실현시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체.

【도면】

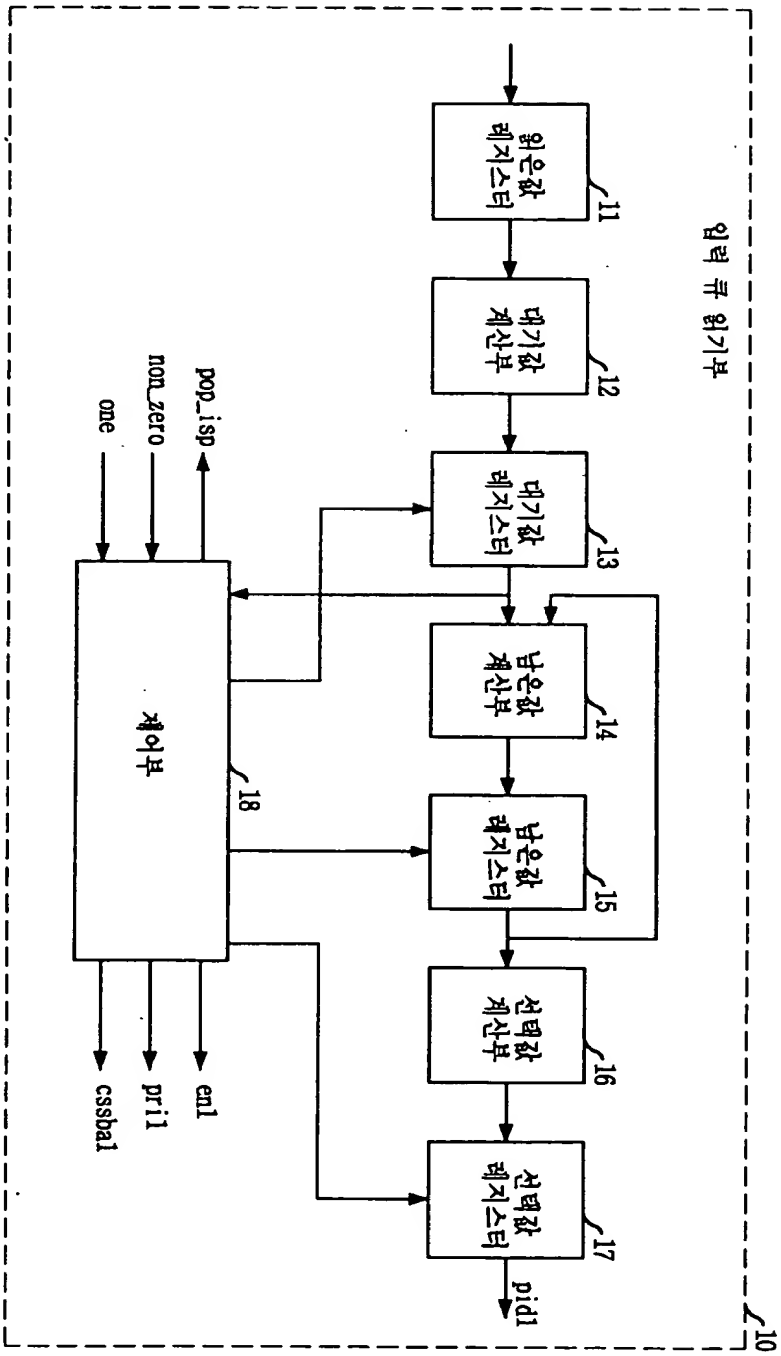
【도 1】


입력 포트 0
입력 포트 1
입력 포트 p-1
입력 서브큐
출력 서브큐0
출력 서브큐1
출력 서브큐2
출력 서브큐n-1

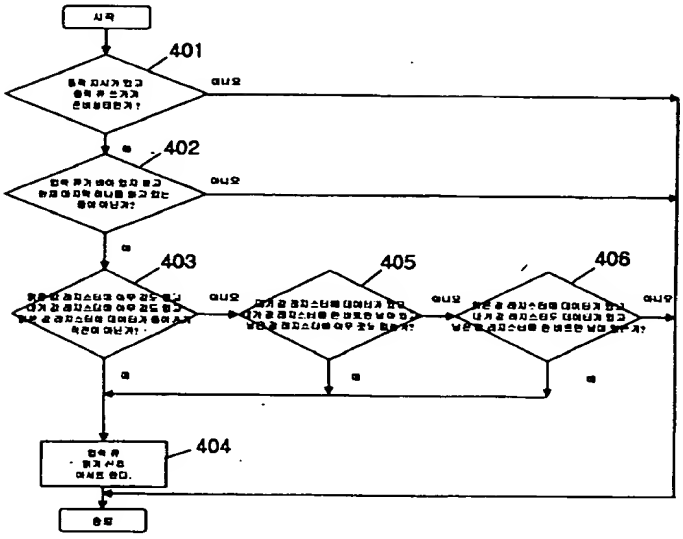
【도 2】


동작지시
초기화 완료
큐 레벨 카운터
10
20
30
pop_isq
nom_zero
cme
sub_cell
입력큐 읽기부
cm1
csaba1
pid1
pri1
큐번호 변환부
cm2
csaba2
pid2
pri2
출력큐 쓰기부
push_esq
esq_addr
csaba
next
pop_esqfa
esqfa
8 비트 비트 맵 형식
3 비트 코딩형식

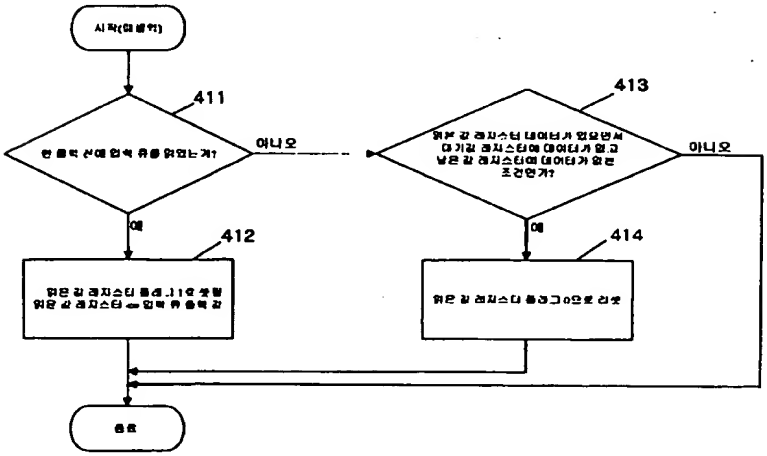
【도 3】


입력 큐 읽기부
읽은값 레지스터
11
대기값 계산부
12
대기값 레지스터
13
남은값 계산부
14
남은값 레지스터
15
선택값 계산부
16
선택값 레지스터
17
pid1
10
제어부
18
pop_isp
non_zero
one
cssba1
pri1
en1

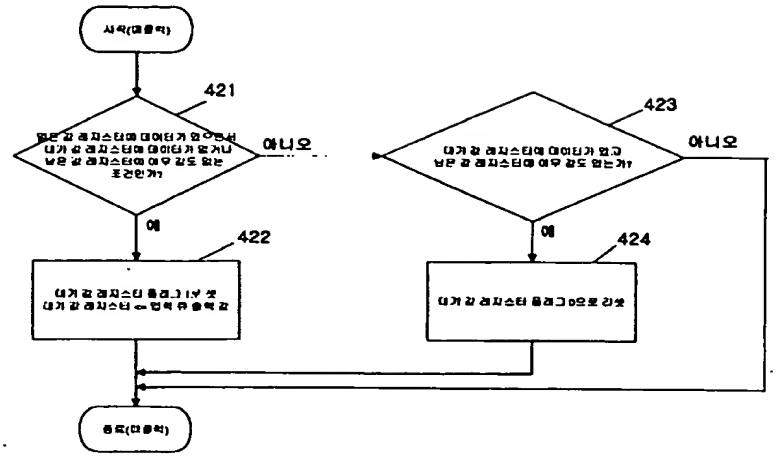
【도 4a】



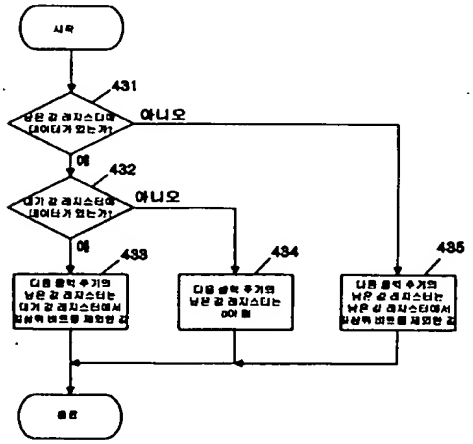
【도 4b】



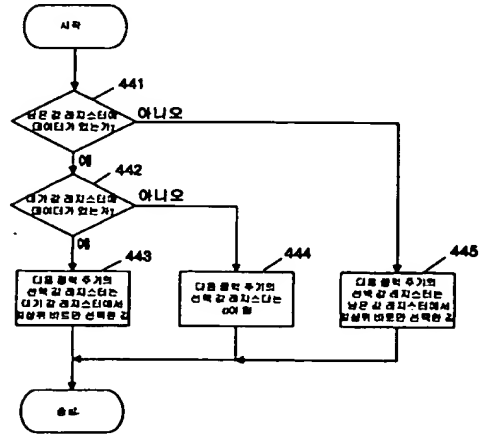
【도 4c】


421
423
아니오
아니오
예
예
422
424

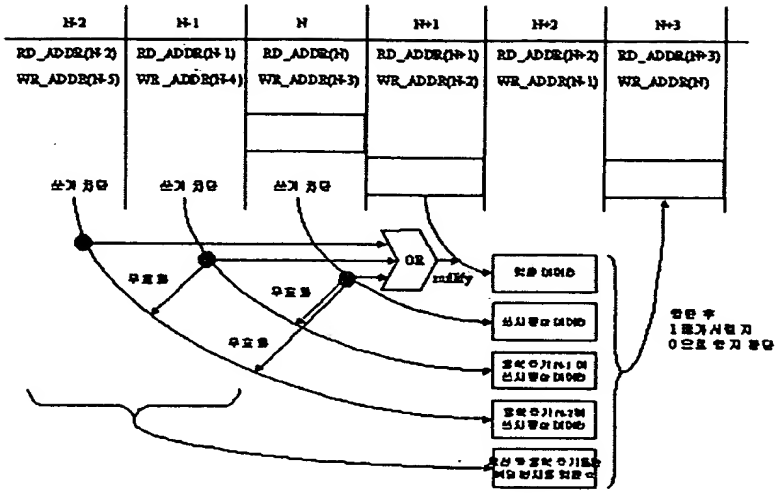
【도 4d】


431
아니오
예
432
아니오
예
433
434
435

【도 4e】


441
아니오
예
442
아니오
예
443
444
445

【도 5】


N-2
N-1
N
N+1
N+2
N+3
RD_ADDR(N-2)
WR_ADDR(N-5)
RD_ADDR(N-1)
WR_ADDR(N-4)
RD_ADDR(N)
WR_ADDR(N-3)
RD_ADDR(N+1)
WR_ADDR(N-2)
RD_ADDR(N+2)
WR_ADDR(N-1)
RD_ADDR(N+3)
WR_ADDR(N)
쓰기 동작
쓰기 동작
쓰기 동작
OR
notify
무효화
무효화
무효화
1 해제시켜 지
0으로 만지 동작